packet **120**, when the actual packet id in the file on the origin server **110** is **142**. The origin server **110** would then send packets **120** through **141** before sending the required packet **142**, which may result in a buffering-timeout problem for clients. In order to alleviate this situation, multiple MBR files associated with the requesting media player with large numbers of streams may be examined in order to determine an upper boundary of a discrepancy between a generated packet id and an actual packet id that can be tolerated. Alternatively, the pre-fetching logic may be modified to increase the prefetching window (e.g., to 20 seconds, as opposed to 10 seconds), which enables the cache **200** to handle extra data received from the origin server **110**.

[**0059**] The foregoing has been a detailed description of an illustrative embodiment of the invention. Various modifications and additions can be made without departing from the spirit and scope of the invention. For example, random time offsets assigned to requesting clients are used to derive "delivery times" in the illustrative embodiment. However, it is expressly contemplated that the invention's teachings and principles relating to choosing a predetermined timing interval based on a client's requested bit rate apply even when the client's derived delivery times do not incorporate random time offsets. In addition, the RTSP and MMS protocols are described herein as exemplary streaming media protocols, although the present invention, in one embodiment, equally applies to other streaming media protocols. Also, a streaming media protocol engine **330** may be configured to check whether a client has security privileges, e.g., according to an access control list (ACL), before the client is permitted to access its requested data stream.

[**0060**] While the illustrative streaming media cache **200** may be a general-purpose computer, it may also be a specialized computer, such as a gaming console, that supplies data streams to one or more clients coupled over a network. Further, it is expressly contemplated that the teachings of this invention can be implemented as software, including a computer-readable medium having program instructions executing on a computer, hardware, firmware, or a combination thereof. Moreover, those skilled in the art will also understand that the teachings set forth herein are not limited to any specific operating system (OS) implementation, and instead may be executed by a wide variety of OS platforms. Accordingly this description is meant to be taken only by way of example and not to otherwise limit the scope of the invention.

1. A storage server comprising:

a streaming media protocol engine to receive a request for a data stream in a first format from a client system;

a streaming media mass storage subsystem to locate the data stream stored by a cache server in a second format; and

a network protocol engine to serve the located data stream to the client system in the first format.

2. The storage server of claim 1, wherein the streaming media mass storage subsystem is further to:

perform a lookup utilizing a lookup key associated with the second format.

3. The storage server of claim 2, wherein the streaming media mass storage subsystem is further to:

generate the lookup key associated with the second format utilizing information associated with the request.

4. The storage server of claim 1, wherein the network protocol engine is further to:

access the data stream stored by the cache server;

access a stored header associated with the data stream;

convert the stored header associated with the data stream to a requested header, the requested header being associated with the first format; and

associate the requested header with the data stream.

5. The storage server of claim 1, wherein the streaming media mass storage subsystem is to:

remove the requested header from the data stream; and

associate the stored header with the data stream;

store the data stream with the associated stored header on a mass storage device associated with the cache server.

6. The storage server of claim 1, wherein the first format is RTSP format.

7. The storage server of claim 1, wherein the second format is MMS format.

8. The storage server of claim 1, wherein the network protocol engine is to:

responsive to the receiving of the request for a data stream in the first format from the client system, identify a version of a media player application associated with the request; and

if the version of the media player application associated with the request is an impermissible version, report a failure to the client system.

9. A method to transmit streaming media, the method comprising:

receiving a request for a data stream in a first format from a client system; and

locating the data stream stored by a cache server in a second format,

serving the located data stream to the client system in the first format.

10. The method of claim 9, wherein the locating of the data stream stored by the cache server comprises:

performing a lookup utilizing a lookup key associated with the second format.

11. The method of claim 10, wherein the locating of the data stream stored by the cache server further comprises:

generating the lookup key associated with the second format utilizing information associated with the request.

12. The method of claim 9, wherein the serving of the located the data stream to the client system in the first format comprises:

accessing the data stream stored by the cache server;

accessing a stored header associated with the data stream;

converting the stored header associated with the data stream to a requested header, the requested header being associated with the first format; and

associating the requested header with the data stream.